Численное решение антагонистической матричной игры

1 Описание

Игра – упрощенная модель конфликта, имеющая четкие правила: варианты действий игроков, объем информации о действиях противника, выигрыш (исход конфликта), к которому приводит совокупность действий игроков.

Игроки – стороны, участвующие в игре: 2 игрока – парная игра, более 2 – множественная игра.

Игра с нулевой суммой (антагонистическая игра) – выигрыш одного из игроков равен проигрышу другого.

Антагонистическая игра называется матричной, если множества стратегий игроков конечны: $X = \{1, ..., m\}$, $Y = \{1, ..., n\}$. При этом принято обозначать стратегию первого игрока через i, стратегию второго через j, а выигрыш первого (проигрыш второго) $F(i, j)$ через a_{ij} . Матрица $A = (a_{ij})_{m \times n}$ называется матрицей игры. Первый игрок выбирает в ней номер строки i, а второй номер столбца j .

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ & & \dots\dots\dots\dots & & \\ a_{m1} & a_{m2} & a_{m3} & \dots & a_{mn} \end{bmatrix}$$

Первый игрок получает максимальный гарантированный (не зависящий от поведения второго игрока) выигрыш, равный цене игры, аналогично, второй игрок добивается минимального гарантированного проигрыша.

Решение антагонистической игры – выбор стратегий, удовлетворяющих условию оптимальности (игрок А должен получать максимальный выигрыш, когда игрок В придерживается своей стратегии, а игрок В должен получать минимальный проигрыш, когда игрок А придерживается своей стратегии).

Под стратегией понимается совокупность правил, определяющих выбор варианта действий при каждом личном ходе игрока в зависимости от сложившейся ситуации

Чистая стратегия – это стратегия, которая применяется с вероятностью 1. В матричной игре чистые стратегии первого игрока – это все $i = 1...m$, второго игрока – все $j = 1...n$.

Смешанная стратегия – указание вероятности, с которой выбирается каждая чистая стратегия.

В смешанных стратегиях $p = (p_1,...,p_m)$ и $q = (q_1,...,q_n)$ - векторы стратегии первого и второго игроков. Если применяется вектор p, то стратегия i выбирается с вероятностью p_i для первого игрока $(i = 1,...,m)$. Аналогично для второго.

$$p_1 + p_2 + ... + p_m = 1$$

$$q_1 + q_2 + ... + q_n = 1$$

Использование ЛП для решения матричной игры - наиболее эффективный прием.

Запишем задачу ЛП

$$a_{11}p_1 + a_{21}p_2 + ... + a_{m1}p_m \geqslant v$$

$$a_{12}p_1 + a_{22}p_2 + ... + a_{m2}p_m \geqslant v$$

$$...$$

$$a_{1n}p_1 + a_{2n}p_2 + ... + a_{mn}p_m \geqslant v$$

$$p_1 + p_2 + ... + p_m = 1, \quad p_i \geqslant 0, \quad i = 1,...,m$$

Поделим все на v и сделаем замену переменных $z_i = p_i/v$.

$$z = (z_1, ..., z_m)$$

$$a_{11}z_1 + a_{21}z_2 + ... + a_{m1}z_m \geqslant 1$$

$$a_{12}z_1 + a_{22}z_2 + ... + a_{m2}z_m \geqslant 1$$

$$...$$

$$a_{1n}z_1 + a_{2n}z_2 + ... + a_{mn}z_m \geqslant 1$$

$$z_1 + z_2 + ... + z_m = (p_1 + ... + p_m)/v = 1/v$$

$v = 1/(z_1^* + ... + z_m^*)$, где $z^* = (z_1^*,...,z_m^*)$ - оптимальное решение задачи ЛП

$$z_1 + z_2 + ... + z_m \to min$$

По z^* находим значение игры и оптимальную смешанную стратегию первого игрока:

$$v = 1/(z_1^* + ... + z_m^*)$$

$$p^* = vz^*$$

Аналогично получаем задачу ЛП для второго игрока

$$w = (w_1, ..., w_n)$$

$$a_{11}w_1 + a_{12}w_2 + ... + a_{1n}w_n \leqslant 1$$

$$a_{21}w_1 + a_{22}w_2 + ... + a_{2n}w_n \leqslant 1$$

...

$$a_{m1}w_1 + a_{m2}w_2 + ... + a_{mn}w_n \leqslant 1$$

$$w_1 + w_2 + ... + w_n \to max$$

$$q^* = vw^*$$

Задача линейного программирования решается с помощью симплекс-метода

Симплекс метод - это метод последовательного перехода от одного базисного решения (вершины многогранника решений) системы ограничений задачи линейного программирования к другому базисному решению до тех пор, пока функция цели не примет оптимального значения (максимума или минимума).

Всякое неотрицательное решение системы ограничений называется допустимым решением.

Пусть имеется система m ограничений с n переменными $(m < n)$.

Допустимым базисным решением является решение, содержащее m неотрицательных основных (базисных) переменных и $n - m$ неосновных. (небазисных, или свободных) переменных. Неосновные переменные в базисном решении равны нулю, основные же переменные, как правило, отличны от нуля, то есть являются положительными числами.

Любые m переменных системы m линейных уравнений с n переменными называются основными, если определитель из коэффициентов при них отличен от нуля. Тогда остальные $n - m$ переменных называются неосновными (или свободными).

Алгоритм симплекс метода

Шаг 1. Привести задачу линейного программирования к канонической форме. Для этого перенести свободные члены в правые части (если среди этих свободных членов окажутся отрицательные, то соответствующее уравнение или неравенство умножить на -1) и в каждое ограничение ввести дополнительные переменные (со знаком ”плюс если в исходном неравенстве знак ”меньше или равно и со знаком ”минус если ”больше или равно”).

Шаг 2. Если в полученной системе m уравнений, то m переменных принять за основные, выразить основные переменные через неосновные и найти соответствующее базисное решение. Если найденное базисное решение окажется допустимым, перейти к допустимому базисному решению.

Шаг 3. Выразить функцию цели через неосновные переменные допустимого базисного решения. Если отыскивается максимум (минимум) линейной формы и в её выражении нет неосновных переменных с отрицательными (положительными) коэффициентами, то критерий оптимальности выполнен и полученное базисное решение является оптимальным - решение окончено. Если при нахождении максимума (минимума) линейной формы в её выражении имеется одна или несколько неосновных переменных с отрицательными (положительными) коэффициентами, перейти к новому базисному решению.

Шаг 4. Из неосновных переменных, входящих в линейную форму с отрицательными (положительными) коэффициентами, выбирают ту, которой соответствует наибольший (по модулю) коэффициент, и переводят её в основные. Переход к шагу 2.

2 Реализация

Считываем матрицу из документа, название которого передаётся в командной строке

На вход функции $nash\_equilibrium()$ подаётся матрица $m \times n$, где m - число стратегий первого игрока, n - второго. Если в матрице есть отрицательный элемент, то находим минимальный элемент и пологаем $a = -min\_el + 1$. Прибавляем a ко всем элементам матрицы.Это нужно, чтобы избежать деления на 0.

Numrows – количество строк

Numcols – количество столбцов

Решаем задачи ЛП с помощью функции linprog() из библиотеки scipy.

Цель функции linprog() - решение задачи линейного программирования.

$linprog(c, A\_ub = None, b\_ub = None, A\_eq = None, b\_eq = None, bounds = None, method =' simplex', callback = None, options = None)$

$$c^T \times x \to min$$

$$A\_ub \times x \leqslant b\_ub$$

$$A\_eq \times x == b\_eq$$

Приводим матрицу и векторы c и b к каноническому виду для функции linprog

c1 – вектор единиц для первой ЗЛП

c2 – вектор минус единиц для второй ЗЛП, т.к. нужно перейти от максимума к минимуму

A2 – просто матрица a для второй ЗЛП

A1 – матрица для первой ЗЛП. Транспонируем исходную матрицу и домножаем на -1, т.к. нужно получить неравенства со знаком $\leqslant$

b2 – вектор единиц для первой ЗЛП

b1 – вектор минус единиц для второй ЗЛП, т.к. нужно поменять знаки, при изменении знака неравенства

Получаем решение для первой и второй ЗЛП

Находим значение игры и оптимальные стратегии игроков

Выводим значение игры и оптимальные стратегии двух игроков.

С помощью библиотеки matplotlib.pyplot рисуем графики.

$fig = plt.figure()$ – создаём фигуру.

Функция $plt.bar(n1, p, width = 0.01, color =' g')$ рисует линии на координатах $n1$ (номер стратегий из p) высотой p .

Функция $plt.plot(n1, p,' go')$ рисует график в координатах $(n1, p)$. В нашем случае ставит точки.

$plt.title$ – пишет заголовок.

Аналогично для стратегий второго игрока

3 Библиотеки

Numpy

Это библиотека языка Python, добавляющая поддержку больших многомерных массивов и матриц, вместе с большой библиотекой математических функций для операций с этими массивами.

Scipy

Библиотека языка Python, предназначенная для выполнения научных и инженерных расчётов.

В нашем случае используется функция linprog() из пакета scipy.optimize, который содержит в себе множество алгоритмов оптимизации.

Matplotlib.pyplot Библиотека на Python для визуализации данных двумерной графикой (3D графика также поддерживается). С помощью неё можно строить графики, показывать статистику, а также рисовать диаграммы.